

УДК 004.4
ББК 32.372
X20

Харенслак Б., де Руйтер Дж.

X20 **Apache Airflow и конвейеры обработки данных / пер. с англ. Д. А. Беликова. – М.: ДМК Пресс, 2022. – 502 с.: ил.**

ISBN 978-5-97060-970-5

Конвейеры обработки данных управляют потоком данных с момента их первоначального сбора до консолидации, очистки, анализа, визуализации и многого другого. Эта книга научит вас создавать и сопровождать эффективные конвейеры обработки данных с использованием платформы Apache Airflow.

Те, кто мало знаком с Airflow, получают базовое представление о принципах работы этой платформы в I части книги. Далее обсуждаются такие темы, как создание собственных компонентов, тестирование, передовые практики и развертывание, – эти главы можно читать в произвольном порядке в зависимости от конкретных потребностей читателя.

Издание предназначено для специалистов по DevOps, обработке и хранению данных, машинному обучению, а также системных администраторов с навыками программирования на Python.

УДК 004.4
ББК 32.372

Original English language edition published by Manning Publications USA. Russian-language edition copyright © 2021 by ДМК Press. All rights reserved.

Все права защищены. Любая часть этой книги не может быть воспроизведена в какой бы то ни было форме и какими бы то ни было средствами без письменного разрешения владельцев авторских прав.

Оглавление

Часть I ■ ПРИСТУПАЕМ К РАБОТЕ	25
1 ■ Знакомство с Apache Airflow.....	27
2 ■ Анатомия ОАГ.....	46
3 ■ Планирование в Airflow.....	67
4 ■ Создание шаблонов задач с использованием контекста Airflow	89
5 ■ Определение зависимостей между задачами.....	114
Часть II ■ ЗА ПРЕДЕЛАМИ ОСНОВ	144
6 ■ Запуск рабочих процессов	146
7 ■ Обмен данными с внешними системами	166
8 ■ Создание пользовательских компонентов	190
9 ■ Тестирование	222
10 ■ Запуск задач в контейнерах	259
Часть III ■ AIRFLOW НА ПРАКТИКЕ	294
11 ■ Лучшие практики.....	295
12 ■ Эксплуатация Airflow в промышленном окружении	324
13 ■ Безопасность в Airflow	369
14 ■ Проект: поиск самого быстрого способа передвижения по Нью-Йорку	393
Часть IV ■ ОБЛАКО	415
15 ■ Airflow и облако	417
16 ■ Airflow и AWS.....	426
17 ■ Airflow и Azure.....	446
18 ■ Airflow в GCP.....	465

Содержание

Предисловие	14
Благодарности	16
О книге	18
Об авторах	23
Об иллюстрации на обложке	24

Часть I ПРИСТУПАЕМ К РАБОТЕ..... 25

1 Знакомство с Apache Airflow	27
1.1 Знакомство с конвейерами обработки данных	28
1.1.1 Конвейеры обработки данных как графы	29
1.1.2 Выполнение графа конвейера	30
1.1.3 Графы конвейеров и последовательные сценарии	32
1.1.4 Запуск конвейера с помощью диспетчеров рабочих процессов	33
1.2 Представляем Airflow	35
1.2.1 Определение конвейеров в коде (Python) гибким образом	35
1.2.2 Планирование и выполнение конвейеров	36
1.2.3 Мониторинг и обработка сбоев	39
1.2.4 Инкрементальная загрузка и обратное заполнение	41
1.3 Когда использовать Airflow	42
1.3.1 Причины выбрать Airflow	42
1.3.2 Причины не выбирать Airflow	43
1.4 Остальная часть книги	44
Резюме	44

2 Анатомия ОАГ	46
2.1 Сбор данных из множества источников	46
2.1.1 Изучение данных	47
2.2 Пишем наш первый ОАГ	48
2.2.1 Задачи и операторы	52
2.2.2 Запуск произвольного кода на Python	53

2.3	Запуск ОАГ в Airflow	56
2.3.1	Запуск Airflow в окружении Python.....	56
2.3.2	Запуск Airflow в контейнерах Docker	57
2.3.3	Изучаем пользовательский интерфейс Airflow	58
2.4	Запуск через равные промежутки времени	62
2.5	Обработка неудачных задач	64
	Резюме	66

3	Планирование в Airflow	67
3.1	Пример: обработка пользовательских событий	68
3.2	Запуск через равные промежутки времени	69
3.2.1	Определение интервалов	70
3.2.2	Интервалы на основе Cron	71
3.2.3	Частотные интервалы	73
3.3	Инкрементная обработка данных.....	74
3.3.1	Инкрементное извлечение событий.....	74
3.3.2	Динамическая привязка ко времени с использованием дат выполнения	75
3.3.3	Разделение данных	77
3.4	Даты выполнения	80
3.4.1	Выполнение работы с фиксированными интервалами.....	80
3.5	Использование обратного заполнения.....	82
3.5.1	Назад в прошлое.....	82
3.6	Лучшие практики для проектирования задач	84
3.6.1	Атомарность.....	84
3.6.2	Идемпотентность	86
	Резюме	87

4	Создание шаблонов задач с использованием контекста Airflow	89
4.1	Проверка данных для обработки с помощью Airflow.....	90
4.1.1	Определение способа загрузки инкрементальных данных	90
4.2	Контекст задачи и шаблонизатор Jinja	92
4.2.1	Создание шаблонов аргументов оператора	92
4.2.2	Что доступно для создания шаблонов?	95
4.2.3	Создание шаблона для PythonOperator	97
4.2.4	Предоставление переменных PythonOperator	102
4.2.5	Изучение шаблонных аргументов.....	104
4.3	Подключение других систем	105
	Резюме	113

5	Определение зависимостей между задачами.....	114
5.1	Базовые зависимости.....	115
5.1.1	Линейные зависимости	115
5.1.2	Зависимости «один-ко-многим» и «многие-к-одному»	116
5.2	Ветвление.....	119
5.2.1	Ветвление внутри задач.....	119

5.2.2	Ветвление внутри ОАГ	121
5.3	Условные задачи	126
5.3.1	Условия в задачах	126
5.3.2	Делаем задачи условными	127
5.3.3	Использование встроенных операторов	129
5.4	Подробнее о правилах триггеров	130
5.4.1	Что такое правило триггеров?	130
5.4.2	Эффект неудач	131
5.4.3	Другие правила	132
5.5	Обмен данными между задачами	133
5.5.1	Обмен данными с помощью XCom	134
5.5.2	Когда (не) стоит использовать XCom	137
5.5.3	Использование настраиваемых XCom-бэкендов	137
5.6	Связывание задач Python с помощью Taskflow API	138
5.6.1	Упрощение задач Python с помощью Taskflow API	139
5.6.2	Когда (не) стоит использовать Taskflow API	141
	Резюме	143

Часть II ЗА ПРЕДЕЛАМИ ОСНОВ 144

6	Запуск рабочих процессов	146
6.1	Опрос условий с использованием сенсоров	147
6.1.1	Опрос пользовательских условий	150
6.1.2	Использование сенсоров в случае сбоя	152
6.2	Запуск других ОАГ	155
6.2.1	Обратное заполнение с помощью оператора TriggerDagRunOperator	159
6.2.2	Опрос состояния других ОАГ	159
6.3	Запуск рабочих процессов с помощью REST API и интерфейса командной строки	163
	Резюме	165

7	Обмен данными с внешними системами	166
7.1	Подключение к облачным сервисам	167
7.1.1	Установка дополнительных зависимостей	168
7.1.2	Разработка модели машинного обучения	169
7.1.3	Локальная разработка с использованием внешних систем	174
7.2	Перенос данных из одной системы в другую	182
7.2.1	Реализация оператора PostgresToS3Operator	184
7.2.2	Привлекаем дополнительные ресурсы для тяжелой работы	187
	Резюме	189

8	Создание пользовательских компонентов	190
8.1	Начнем с PythonOperator	191
8.1.1	Имитация API для рейтинга фильмов	191
8.1.2	Получение оценок из API	194
8.1.3	Создание фактического ОАГ	197

8.2	Создание собственного хука	199
8.2.1	Создание собственного хука	200
8.2.2	Создание ОАГ с помощью <i>MovielensHook</i>	206
8.3	Создание собственного оператора	208
8.3.1	Определение собственного оператора	208
8.3.2	Создание оператора для извлечения рейтингов	210
8.4	Создание нестандартных сенсоров	213
8.5	Упаковка компонентов	216
8.5.1	Создание пакета <i>Python</i>	217
8.5.2	Установка пакета	219
	Резюме	220

9 Тестирование

9.1	Приступаем к тестированию	223
9.1.1	Тест на благонадежность ОАГ	223
9.1.2	Настройка конвейера непрерывной интеграции и доставки	230
9.1.3	Пишем модульные тесты	232
9.1.4	Структура проекта <i>Pytest</i>	233
9.1.5	Тестирование с файлами на диске	238
9.2	Работа с ОАГ и контекстом задачи в тестах	241
9.2.1	Работа с внешними системами	246
9.3	Использование тестов для разработки	254
9.3.1	Тестирование полных ОАГ	257
9.4	Эмулируйте промышленное окружение с помощью <i>Whirl</i>	257
9.5	Создание окружений	258
	Резюме	258

10 Запуск задач в контейнерах

10.1	Проблемы, вызываемые множеством разных операторов	260
10.1.1	Интерфейсы и реализации операторов	260
10.1.2	Сложные и конфликтующие зависимости	261
10.1.3	Переход к универсальному оператору	261
10.2	Представляем контейнеры	262
10.2.1	Что такое контейнеры?	263
10.2.2	Запуск нашего первого контейнера <i>Docker</i>	264
10.2.3	Создание образа <i>Docker</i>	265
10.2.4	Сохранение данных с использованием томов	267
10.3	Контейнеры и <i>Airflow</i>	270
10.3.1	Задачи в контейнерах	270
10.3.2	Зачем использовать контейнеры?	270
10.4	Запуск задач в <i>Docker</i>	272
10.4.1	Знакомство с <i>DockerOperator</i>	272
10.4.2	Создание образов для задач	274
10.4.3	Создание ОАГ с задачами <i>Docker</i>	277
10.4.4	Рабочий процесс на базе <i>Docker</i>	280
10.5	Запуск задач в <i>Kubernetes</i>	281
10.5.1	Представляем <i>Kubernetes</i>	282
10.5.2	Настройка <i>Kubernetes</i>	283
10.5.3	Использование <i>KubernetesPodOperator</i>	286

10.5.4	Диагностика проблем, связанных с Kubernetes	290
10.5.5	Отличия от рабочих процессов на базе Docker	292
Резюме	293

Часть III AIRFLOW НА ПРАКТИКЕ 294

11 Лучшие практики 295

11.1	Написание чистых ОАГ	296
11.1.1	Используйте соглашения о стилях	296
11.1.2	Централизованное управление учетными данными	300
11.1.3	Единообразно указывайте детали конфигурации	301
11.1.4	Избегайте вычислений в определении ОАГ	304
11.1.5	Используйте фабричные функции для генерации распространенных шаблонов	306
11.1.6	Группируйте связанные задачи с помощью групп задач	310
11.1.7	Создавайте новые ОАГ для больших изменений	312
11.2	Проектирование воспроизводимых задач	312
11.2.1	Всегда требуйте, чтобы задачи были идемпотентными	312
11.2.2	Результаты задачи должны быть детерминированными	313
11.2.3	Проектируйте задачи с использованием парадигмы функционального программирования	313
11.3	Эффективная обработка данных	314
11.3.1	Ограничьте объем обрабатываемых данных	314
11.3.2	Инкрементальная загрузка и обработка	316
11.3.3	Кешируйте промежуточные данные	317
11.3.4	Не храните данные в локальных файловых системах	318
11.3.5	Переложите работу на внешние/исходные системы	318
11.4	Управление ресурсами	319
11.4.1	Управление параллелизмом с помощью пулов	319
11.4.2	Обнаружение задач с длительным временем выполнения с помощью соглашений об уровне предоставления услуг и оповещений	321
Резюме	322

12 Эксплуатация Airflow в промышленном окружении 324

12.1	Архитектура Airflow	325
12.1.1	Какой исполнитель мне подходит?	327
12.1.2	Настройка базы метаданных для Airflow	328
12.1.3	Присмотримся к планировщику	330
12.2	Установка исполнителей	334
12.2.1	Настройка SequentialExecutor	335
12.2.2	Настройка LocalExecutor	335
12.2.3	Настройка CeleryExecutor	336
12.2.4	Настройка KubernetesExecutor	339
12.3	Работа с журналами всех процессов Airflow	347
12.3.1	Вывод веб-сервера	347
12.3.2	Вывод планировщика	348

12.3.3	Журналы задач	349
12.3.4	Отправка журналов в удаленное хранилище	350
12.4	Визуализация и мониторинг метрик Airflow	350
12.4.1	Сбор метрик из Airflow	351
12.4.2	Настройка Airflow для отправки метрик	353
12.4.3	Настройка Prometheus для сбора метрик	353
12.4.4	Создание дашбордов с Grafana	356
12.4.5	Что следует мониторить?	358
12.5	Как получить уведомление о невыполненной задаче	360
12.5.1	Оповещения в ОАГ и операторах	360
12.5.2	Определение соглашений об уровне предоставления услуги	362
12.6	Масштабируемость и производительность	364
12.6.1	Контроль максимального количества запущенных задач	365
12.6.2	Конфигурации производительности системы	366
12.6.3	Запуск нескольких планировщиков	367
	Резюме	368

13	Безопасность в Airflow	369
13.1	Обеспечение безопасности веб-интерфейса Airflow	370
13.1.1	Добавление пользователей в интерфейс RBAC	371
13.1.2	Настройка интерфейса RBAC	374
13.2	Шифрование хранимых данных	375
13.2.1	Создание ключа Fernet	375
13.3	Подключение к службе LDAP	377
13.3.1	Разбираемся с LDAP	378
13.3.2	Извлечение пользователей из службы LDAP	380
13.4	Шифрование трафика на веб-сервер	381
13.4.1	Разбираемся с протоколом HTTP	381
13.4.2	Настройка сертификата для HTTPS	384
13.5	Извлечение учетных данных из систем управления секретами	388
	Резюме	392

14	Проект: поиск самого быстрого способа передвижения по Нью-Йорку	393
14.1	Разбираемся с данными	396
14.1.1	Файловый ресурс Yellow Cab	397
14.1.2	REST API Citi Bike	397
14.1.3	Выбор плана подхода	399
14.2	Извлечение данных	400
14.2.1	Скачиваем данные по Citi Bike	400
14.2.2	Загрузка данных по Yellow Cab	402
14.3	Применение аналогичных преобразований к данным	405
14.4	Структурирование конвейера обработки данных	410
14.5	Разработка идемпотентных конвейеров обработки данных	411
	Резюме	414

Часть IV ОБЛАКО415

15 *Airflow и облако*417

- 15.1 Проектирование стратегий (облачного) развертывания418
- 15.2 Операторы и хуки, предназначенные для облака420
- 15.3 Управляемые сервисы421
 - 15.3.1 *Astronomer.io*421
 - 15.3.2 *Google Cloud Composer*422
 - 15.3.3 *Amazon Managed Workflows for Apache Airflow*423
- 15.4 Выбор стратегии развертывания423
- Резюме425

16 *Airflow и AWS*426

- 16.1 Развертывание Airflow в AWS426
 - 16.1.1 Выбор облачных сервисов427
 - 16.1.2 Проектирование сети428
 - 16.1.3 Добавление синхронизации ОАГ430
 - 16.1.4 Масштабирование с помощью *CeleryExecutor*430
 - 16.1.5 Дальнейшие шаги432
- 16.2 Хуки и операторы, предназначенные для AWS432
- 16.3 Пример использования: бессерверное ранжирование фильмов с AWS Athena434
 - 16.3.1 Обзор434
 - 16.3.2 Настройка ресурсов435
 - 16.3.3 Создание ОАГ438
 - 16.3.4 Очистка445
- Резюме445

17 *Airflow и Azure*446

- 17.1 Развертывание Airflow в Azure446
 - 17.1.1 Выбор сервисов447
 - 17.1.2 Проектирование сети448
 - 17.1.3 Масштабирование с помощью *CeleryExecutor*449
 - 17.1.4 Дальнейшие шаги450
- 17.2 Хуки и операторы, предназначенные для Azure451
- 17.3 Пример: бессерверное ранжирование фильмов с Azure Synapse452
 - 17.3.1 Обзор452
 - 17.3.2 Настройка ресурсов453
 - 17.3.3 Создание ОАГ457
 - 17.3.4 Очистка463
- Резюме464

18 *Airflow в GCP*465

- 18.1 Развертывание Airflow в GCP465
 - 18.1.1 Выбор сервисов466
 - 18.1.2 Развертывание в GKE с помощью Helm468

18.1.3	Интеграция с сервисами Google.....	471
18.1.4	Проектирование сети.....	472
18.1.5	Масштабирование с помощью CeleryExecutor	473
18.2	Хуки и операторы, предназначенные для GCP	476
18.3	Пример использования: бессерверный рейтинг фильмов в GCP.....	481
18.3.1	Загрузка в GCS.....	481
18.3.2	Загрузка данных в BigQuery.....	483
18.3.3	Извлечение рейтингов, находящихся в топe	485
	Резюме	488
Приложение А	Запуск примеров кода	490
Приложение В	Структуры пакетов Airflow 1 и 2	494
Приложение С	Сопоставление метрик в Prometheus	498
	Предметный указатель	500